Taming Large Models with Hawk and NeoEMF

A. García-Domínguez, D. S. Kolovos, K. Barmpis, G. Daniel, G. Sunyé

MoDELS'2018, 14–19 October 2018

Table of contents

1. Introduction

2. Hawk

3. NeoEMF

4. Mogwaï

5. Wrap-up

Introduction

Who are we? — Hawk team

Antonio (Lecturer, Aston University)

- Hawk project lead
- Eclipse Epsilon committer

Konstantinos (Research Associate, U. of York)

- Hawk initiator (core developer)
- Created most core components

Dimitris (Professor, University of York)

- Hawk initiator (MONDO WP lead)
- Eclipse Epsilon project lead

Who are we? — NeoEMF team

Gwendal (Post-doc, SOM Research Lab)

- NeoEMF core developer
- Mogwaï project lead

Gerson (Associate Professor, U. of Nantes)

- NeoEMF initiator
- AtlanMod project lead

Motivation: monolithic XMI models do not scale

"[Lack of] Scalability is what is holding back a number of potential adopters" [BK13]

"One can easily observe that scalability is the most critical of today's (and tomorrow's) challenges" [MDBS09]

Model	XMI (MB)	Avg. memory (MB)	Max memory (MB)
set1	26.59	53	135
set2	270.12	437	840
set3	597.67	849	1798
set4	645.53	949	1910

Table 1: Querying GraBaTs'09 Java models to find singletons

What to do?

Option 1: break up into fragments

- Code is broken up into modules — do the same with models
- Fragmented file-based models play well with traditional VCS
- Global queries may still require loading all fragments!
- **Hawk** is our solution: indexes the fragments into a NoSQL database, queries the database instead of the fragments

Option 2: stop using files

- Obviously not new: CDO has been doing it for years
- However, relational DBs are not the only option
- **NeoEMF** can replace traditional XMI persistence with NoSQL approaches: graph databases, key-value stores...

Basic concepts about NoSQL

NoSQL = not only SQL

- Generally, non-relational databases
- Some improve availability/performance by relaxing ACID
- Some are focused on specific analyses (e.g. social graphs)

Common types

- **Key-value stores** implement an associative array to quickly fetch records given a tuple-based identifier (e.g. RocksDB, Redis).
- **Document stores** keep collections of heterogeneous docs retrievable by key, supporting hierarchies + querying by field (e.g. MongoDB).
- **Graph databases** store nodes, edges and their fields (e.g. Neo4j, OrientDB). Many support indexing and have query languages.
- **Tuple stores** store subject-predicate-object triples (e.g. RDF4J, Jena). Very often used in Semantic Web approaches.

Some considerations before we start

NoSQL = purpose-specific databases

- Can provide better capabilities than RDBMS, but require careful choice + benchmarking + tuning — luckily, we did this for you :-)
- Technology/release/API all impact performance:
  - Hawk Neo4j backend still in v2.0.5: later v2.x *were slower*, as Neo4j optimized for different requirements
  - Hawk OrientDB v2.2.30 backend avoids *slower* official OrientDB graph API, and uses the core document API instead

Consider your requirements

- Do you need faster queries?
- Do you need faster loading/saving? If so:
  - Can you break up your files into fragments?
  - Can you save to disk only the fragments that changed?

We will revisit these during the wrap-up.

Done with intro — let's start with Hawk!

References

Konstantinos Barmpis and Dimitrios S. Kolovos.
Hawk: towards a scalable model indexing architecture.
In *Proceedings of the Workshop on Scalability in Model Driven Engineering*, BigMDE '13, pages 6:1–6:9, New York, NY, USA, June 2013. ACM.

Alix Mougenot, Alexis Darrasse, Xavier Blanc, and Michèle Soria.
Uniform Random Generation of Huge Metamodel Instances.
In *Proceedings of ECMDA-FA '09*, pages 130–145, Berlin, Heidelberg, 2009. Springer-Verlag.